

(19)



Eur päisches Patentamt

European Patent Office

Office européen des brevets



(11)

EP 0 969 366 A1

(12)

EUROPEAN PATENT APPLICATION

(43) Date of publication:
05.01.2000 Bulletin 2000/01

(51) Int. Cl.⁷: **G06F 9/46, G06F 1/00**

(21) Application number: 99202070.1

(22) Date of filing: 26.06.1999

(84) Designated Contracting States:
**AT BE CH CY DE DK ES FI FR GB GR IE IT LI LU
MC NL PT SE**
Designated Extension States:
AL LT LV MK RO SI

(72) Inventors:
• **Lipkin, Efrem**
Berkeley, CA 94703 (US)
• **Goldstein, Theodore C.**
Palo Alto, CA 94306 (US)

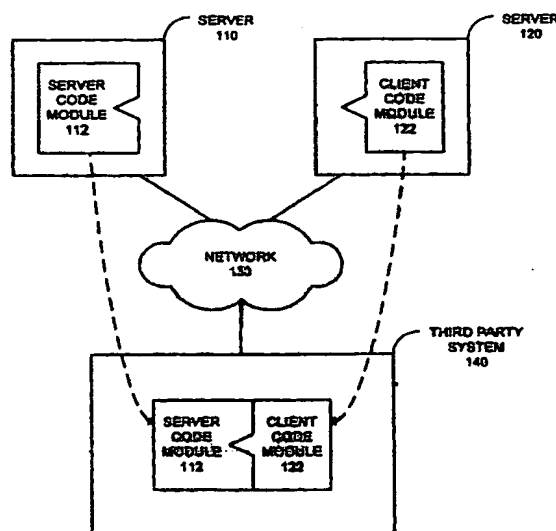
(30) Priority: 29.06.1998 US 106567

(74) Representative:
Hanna, Peter William Derek et al
Tomkins & Co.,
5 Dartmouth Road
Dublin 6 (IE)

(71) Applicant:
SUN MICROSYSTEMS, INC.
Palo Alto, California 94303 (US)

(54) Controlling access to services between modular applications

(57) The present invention provides a method and an apparatus for providing a first computer program module (122) with the ability to access a service from a second computer program module (112). The method includes receiving the first computer program module (12) for example, at a third party computer system (140), and determining whether the first computer program module has been digitally signed by an authority (204) having power to confer access for the service. If so, the method provides the first computer program module (122) with access to the service. A variation on this embodiment includes verifying that the first computer program module (122) includes a chain of certificates establishing a chain of authorisation for the service. This verification process includes verifying that a first certificate in the chain is signed by an entity (400) that is originally authorised to confer access for the service, and verifying that subsequent certificates in the chain are signed by entities (430,450) that have been delegated authorisation to confer access for the service. In a further variation on the above embodiment, the act of providing the first computer program module with access to the service, includes providing the first computer program module (122) with a permit that allows the first computer program module (122) to perform a restricted set of operations on the service.

**FIG. 1****EP 0 969 366 A1**

Description

BACKGROUND

[0001] The present invention relates to protection mechanisms in computer systems. More specifically, the present invention relates to a method and an apparatus for controlling access to services.

[0002] Programming languages such as the Java™ programming language (developed by SUN Microsystems, Inc. of Palo Alto, California) and associated supporting interfaces presently provide a reliable and secure infrastructure to support the transfer of an application across a computer network, and to run the application on a wide range of computing platforms. Because of developments such as Java, it is becoming increasingly common to load an application, such as a Java applet, from a remote server onto a local machine, and to execute the application on the local machine.

[0003] However, present computing systems are not designed to allow computer applications from different vendors to interact with each other in a controlled way so that the applications can work together to accomplish a given task. In particular, these systems do not facilitate sharing of data and functions. For example, it may be useful for a tax application to access capital gains information from a home brokerage application. However, the home brokerage application needs to protect the privacy of the customer's portfolio. Hence, the tax application cannot be given unrestricted access portfolio data from the home brokerage application.

[0004] Additionally, software vendors may want to enforce contractual arrangements between complementary applications. For example, a home brokerage application may want to tap into historical pricing information supplied by an application from a financial institution. This arrangement would be facilitated if the vendor of the home brokerage application would establish a contractual arrangement with the financial institution that allows the home brokerage application to access the historical pricing information.

[0005] Unfortunately, present computing systems lack any mechanism for facilitating and controlling access to services provided by other applications. In particular, with present systems it is not possible to identify applications that have been granted rights to access services from other applications, nor to control what services a given application can have performed.

SUMMARY

[0006] The present invention provides a method and an apparatus for providing a first computer program module with the ability to access a service from a second computer program module. The method includes receiving the first computer program module -- for example, at a third party computer system, and determining whether the first computer program module has

been digitally signed by an authority having power to confer access for the service. If so, the method provides the first computer program module with access to the service. A variation on this embodiment includes verifying that the first computer program module includes a chain of certificates establishing a chain of authorization for the service. This verification process includes verifying that a first certificate in the chain is signed by an entity that is originally authorized to confer access for the service, and verifying that subsequent certificates in the chain are signed by entities that have been delegated authorization to confer access for the service.

[0007] In a further variation on the above embodiment, the act of providing the first computer program module with access to the service, includes providing the first computer program module with a permit that allows the first computer program module to perform a restricted set of operations on the service.

[0008] In another variation on the above embodiment, the first computer program module and the second computer program module can interact with each other on a third party computer system. In this case, the first computer program module is transferred from a first server to the third party system, and the second computer program module is transferred from a second server to the third party system. This allows the first computer program module and the second computer program module to interact with each other on the third party system.

BRIEF DESCRIPTION OF THE FIGURES

[0009]

FIG. 1 illustrates a number of computer nodes coupled together through a network 130 in accordance with an embodiment of the present invention.

FIG. 2 illustrates the process of receiving access to a service in accordance with an embodiment of the present invention.

FIG. 3 illustrates part of the structure of client code module 122 from FIG. 1 in accordance with an embodiment of the present invention.

FIG. 4 illustrates how authority to access a service is transferred between different entities using a chain of certificates in accordance with an embodiment of the present invention.

FIG. 5 is a flow chart illustrating how authorization to access a service is propagated, and is ultimately used gain access to the service, in accordance with an embodiment of the present invention.

DETAILED DESCRIPTION

[0010] The following description is presented to enable any person skilled in the art to make and use the invention, and is provided in the context of a particular application and its requirements. Various modifications

to the disclosed embodiments will be readily apparent to those skilled in the art, and the general principles defined herein may be applied to other embodiments and applications without departing from the spirit and scope of the present invention. Thus, the present invention is not intended to be limited to the embodiments shown, but is to be accorded the widest scope consistent with the principles and features disclosed herein.

[0011] For purposes of this detailed disclosure the following terminology is used. (1) A "Java Archive file" can be a file containing modular bodies of Java™ code in addition to resources, such as graphics or audio files. (2) A "computer readable storage medium" can be any device or medium that can store code and/or data for use by a computer system. This includes, but is not limited to, magnetic and optical storage devices such as disk drives, magnetic tape, CDs (compact discs) and DVDs (digital video discs), or alternatively, computer instruction signals embodied in a carrier wave. (3) A "computer program module" can be a module including a collection of instructions that can be executed by a computer. These instructions may comprise an entire computer program, or merely a piece of a computer program. A computer program module often exists in a form that facilitates downloading onto a computer system across a computer network. For example, a computer program module may take the form of a Java™ Applet. (4) An "entity" can be a human being, a computer program or a computer system, that has the power to confer access rights for a service, and optionally the ability to delegate such power to other entities. (5) A "service" can include a single service or a plurality of services. Therefore, the act of conferring access for a service can also confer access to a plurality of services.

Computer System

[0012] FIG. 1 illustrates a number of computer nodes coupled together through a network 130 in accordance with an embodiment of the present invention. In FIG. 1, servers 110 and 120 are coupled to third party system 140 through network 130. A computer node can be any computation device that can be coupled to a computer network. A computer node may include, but is not limited to, a personal computer, a workstation, a mainframe computer, a portable computer or a device controller. Network 130 generally refers to any type of wire or wireless link between computers, including, but not limited to, a local area network, a wide area network, or a combination of networks. In one embodiment of the present invention, network 130 includes the Internet. Servers 110 and 120 can be any nodes on a computer network including a mechanism for servicing requests from a client for computational or data storage resources. Third party system 140 may be any node a computer network communicating with servers 110 and 120 that is able to download code and/or data from servers 110 and 120.

[0013] In the embodiment illustrated in FIG. 1, server 110 contains server code module 112, and server 120 contains client code module 122. For purposes of this detailed disclosure, a server code module is a module including code that provides a service to a client code module, and a client code module is a module including code that requests a service from a server code module. Server code module 112 and client code module 122 include modular pieces of code that can operate together on third party system 140. The dashed lines on FIG. 1 represent server code module 112 and client code module 122 being downloaded onto third party system 140 across network 130. This downloading process can take place in a number of ways. In one embodiment of the present invention, server 110 includes a web site that can be accessed by a user on third party system 140 to download server code module 112 onto third party system 140. Correspondingly, server 120 includes a web site that can be accessed by a user on third party system 140 to download client code module 122 into third party system 140. In another embodiment, server code module 112 and client code module 122 are not downloaded across network 130. Instead, they are transferred from servers 110 and 120, respectively, to third party system 140 by way of computer storage media, such as a computer disk.

[0014] Once server code module 112 and client code module 122 are located on third party system 140, they can be integrated to work together as is illustrated in FIG. 1. For example, in providing a service to client code module 122, server code module 112 might retrieve data from a database for client code module 122. Alternatively, server code module 112 might perform a computational operation for client code module 122. This integration process may involve determining whether client code module 122 has been conferred the right to access services from server code module 112. In the reverse direction, this process may involve determining whether server code module 112 has been conferred the right to access services from client code module 122.

Access Model

[0015] FIG. 2 illustrates the process of accessing a service in accordance with an embodiment of the present invention. FIG. 2 illustrates interactions between server gate 202, system 204 and client code module 122 (from FIG. 1). Server gate 202 includes an access mechanism that controls access to services provided by server code module 112 (from FIG. 1). In one embodiment of the present invention, server gate 202 is located within server code module 112 on third party system 140. In another embodiment, server gate 202 is located within server 110 itself, and is accessed via communications across network 130. System 204 includes a mechanism for establishing that client code module 122 is properly authorized to access services

provided by server code module 112. To this end, system 204 is implemented in a number of ways. In one embodiment, system 204 is implemented by code that is part of third party system 140. In another embodiment, system 204 may be implemented as part of server code module 112 within third party system 140.

[0016] The process illustrated in FIG. 2 operates as follows. Client code module 122 is assumed to already exist within third party system 140. In order to access a desired service, client code module 122 requests a "ticket" for a "role" to access a collection of services from server code module 112. (For purposes of this detailed disclosure, a ticket is an object that cannot be forged that indicates that the holder of the object has been signed to use certain services.) A role defines a set of operations to be performed by server code module 112. Certain roles may be more limited than other roles. For example, if server code module 112 maintains a computer file system, one role may include only the operation of reading a file from the file system. Another more powerful role may include the operations of reading, writing and deleting files from the file system.

[0017] In response to the request, system 204 examines client code module 122 to determine if client code module 122 includes proper authorization for the role. In one embodiment of the present invention, this examination includes examining a certificate chain 310 (illustrated in FIG. 3) to ensure that certificate chain 310 has been properly signed by a chain of authorities. This process is described in more detail below with reference to FIGs. 3-5.

[0018] If client code module 122 is properly authorized for the role, system 204 issues a ticket for the role, and this ticket is given to client code module 122. Next, client code module 122 passes the ticket to server gate 202. Server gate 202 checks the ticket to ensure that the ticket is valid. If it is valid, server gate 202 sends a permit for the service to client code module 122. (For purposes of this detailed description, a permit is a proxy or a capability giving a holder of the permit access to a service or a group of services.) This permit allows client code module 122 to access the services defined by the role. In one embodiment of the present invention, this permit is an object defined within an object-oriented programming system. This object allows client code module 122 to perform a set of methods that comprise the role. After the permit is sent, server gate 202 invalidates the ticket, so that it cannot be used again. Since client code module 122 remains in possession of the permit, client code module 122 will be able to access services using the permit, and hence, no longer needs the ticket.

Client Code Module

[0019] FIG. 3 illustrates part of the structure of client code module 122 from FIG. 1 in accordance with an embodiment of the present invention. Client code mod-

ule 122 includes certificate chain 310 and client code 320. Certificate chain 310 includes a chain of certificates that establishes a chain of authorization for the service. The first certificate in the chain is signed by an entity that is originally authorized to confer access for the service, and subsequent certificates in the chain are signed by entities that have been delegated authorization to confer access for the service from preceding entities in the chain.

[0020] For purposes of this detailed disclosure, a certificate is a signed electronic document that certifies that something is true. A certificate typically indicates that someone has ownership of a public key. In the present invention, a certificate can indicate that an entity can have access to services represented by a key. A certificate may include the identity of a signing authority as well as a digital signature produced with a private key (that can be validated with a corresponding public key). For example, one certificate format is defined under the X.509 standard.

[0021] For purposes of this detailed disclosure, a digital signature is a value derived from a file using a secret such that it can be demonstrated that the value was derived using the secret, wherein the secret is known only to the signer. A digital signature may take the form of a message digest produced by the key and appended to the file, or may take the form of a transformation of data within the file using the key. A digital signature may also take the form of a message digest encrypted by the private key of a public key private key cryptography system.

[0022] For example, in the illustrated embodiment certificate chain 310 includes certificate-1 312, certificate-2 314 and certificate-N 316. A server code owner initially starts with a private key zero. In order to pass along authority for a role, the server code owner generates a certificate-1 312 and an associated public key private key pair, the private key being private key one. The server code owner signs certificate-1 312 with private key zero and passes certificate-1 312 along with the corresponding private key one to a first intermediary. The first intermediary generates certificate-2 314 along with a corresponding public key private key pair, including private key two. The first intermediary signs certificate-2 314 with private key one and passes certificate-2 314, along with the associated private key two and all previous certificates in the chain, to a following intermediary. This pattern continues up the chain until a final intermediary signs certificate-N 316 with private key N-1 and passes the certificate-N 316, along with corresponding private key N and all previous certificates in the chain, to a client code owner. The client code owner uses private key N to sign client code 320, and then generates client code module 122, which includes certificate chain 310 and client code 320.

[0023] Hence, client code module 122 includes a verifiable chain of certificates 310 signed by intermediaries from the server code owner to the ultimate client code

owner. Certificate chain 310 can be verified by using the public keys to verify that certificates in the chain are properly signed with their corresponding private keys.

Delegation of Authority

[0024] FIG. 4 illustrates how authority to access a service is delegated between different entities using a chain of certificates in accordance with an embodiment of the present invention. In this example, server company 400 delegates authority to access data associated with a server code module, such as server code module 112 in FIG. 1. This server code module is distributed to various third party systems, and can interact with properly authorized client code modules on these third party systems. Alternatively, the server code module can interact with client code modules on computer systems belonging to either server company 400 or to the owners of the client code modules.

[0025] In the example illustrated in FIG. 4, server company 400 does the following. First, server company 400 generates a public/private key pair, including private key zero. Next, server company 400 generates server code 460, which checks to see that client code modules include a chain of certificates, including a root certificate signed with private key zero. Second, server company 400 generates a certificate and a public/private key pair for each intermediary or client company that it desires to delegate authority to. Third, it sends the certificate signed with private key zero and the private key associated with the certificate to the intermediary or client company. In the illustrated example, server company 400 sends certificate X 404 (signed with private key zero) and private key X 402 to client company X 430. Server company 400 additionally sends certificate Y 420 (signed with private key zero) and private key Y 418 to intermediary Y 450.

[0026] In the example illustrated in FIG. 4, client company X 430 generates certificates and public/private key pairs for each of three projects, and passes the certificates and associated private keys to entities within client company X 430 that are responsible for producing three different client code modules. In particular, client company X 430 passes certificate X1 408 (signed with private key X) and private key X1 406 to project X1 432. Client company X 430 also passes certificate X2 412 (signed with private key X) and private key X2 410 to project X2 434. Client company X 430 additionally passes certificate X3 416 (signed with private key X) and private key X3 414 to project X3 436.

[0027] Next, each project within client company X 430 creates a code module. In particular, project X1 432 creates a code module 438 for project X1. This code module includes a chain of certificates, including certificate X 404 (signed with private key zero) and certificate X1 408 (signed with private key X 402). Code module 438 also includes code (not shown) that is signed with private key X1 406. Project X2 434 creates code module

440 for project X2. This code module includes a chain of certificates, including certificate X 404 (signed with private key zero) and certificate X2 412 (signed with private key X 402). Code module 440 also includes code (not shown) that is signed with private key X2 410. Project X3 436 creates code module 442 for project X3. This code module includes a chain of certificates, including certificate X 404 (signed with private key zero) and certificate X3 416 (signed with private key X 402). Code module 442 also includes code (not shown) that is signed with private key X3 414.

[0028] In the example illustrated in FIG. 4, intermediary Y 450 generates a certificate Z 424 and a public, private key pair, including private key Z 422. Intermediary Y 450 signs certificate Z 424 using private key Y 418 and passes certificate Z 424 (signed with private key Y 418) along with private key Z 422 to client company Z 452.

[0029] Client company Z 452 creates code module 454 for project Z, which includes a chain of certificates, including certificate Y 420 (signed with private key zero) and certificate Z 424 (signed with private key Y 418). Code module 454 also includes code (not shown) that is signed with private key Z 422.

Delegation and Authorization Process

[0030] FIG. 5 is a flow chart illustrating how authorization to access a service is propagated and is ultimately used gain access to the service in accordance with an embodiment of the present invention. The system starts at state 500 and proceeds to state 502. In state 502, server company 400 (from FIG. 4) creates server code 460, which checks for clients being signed with key zero. Key zero is associated with a particular role, which defines a set of services that may be performed in the role. The system next proceeds to state 504. In state 504, server company 400 creates for each client a new public/private key pair and a certificate. The system next proceeds to state 506. In state 506, server company 400 exchanges these certificates and private keys with the clients. This exchange may involve a transfer of money in payment for the use of the service or some other contractual consideration. The system next proceeds to state 508.

[0031] In state 508, each client company optionally generates its own public/private key pairs and matching certificates for each client code module that is to assume the role represented by key zero. This process may be repeated for numerous levels of clients and intermediaries until a final client that owns the client code is reached. The system next proceeds to state 510.

[0032] In state 510, the final client signs the client code with the last key in the chain and packages it with all certificates in the chain. The system next proceeds to state 512. In state 512, client code module 122 is downloaded to a third party system 140, which also loads

server code module 112 from server company 400. The system then proceeds to state 514.

[0033] In state 514, the client code requests access to the service stored by the server code by requesting a ticket for a role from the system. This role specifies certain operations on the service. The system next proceeds to state 516. In state 516, the system checks the validity of the request. This is done by examining all certificates in the chain and the client code to ensure that the certificates and the client code are signed with the proper private keys. This is accomplished by using the corresponding public keys to verify signing by the corresponding private keys. If the request is valid, the system returns a ticket to the client.

[0034] The process of examining the chain of certificates may be carried completely by the server code, or completely by neutral code on the third party system. Alternatively, a portion of the examination can be carried out by the system code and a portion carried out by the neutral code. For example, the neutral code can examine all of the certificates except the first certificate, and the server code can examine the first certificate to verify that it is signed by private key zero. The system next proceeds to state 518. In state 518, the client code passes the ticket to server gate 202 (as was described above with reference to FIG. 2). Server gate 202 checks the validity of the ticket, and if valid, server gate 202 sends to the client code a permit to access the service through the role. The system next proceeds to state 522, which is an end state. The above-described process is repeated for each new server code module or client code module that the system desires to create.

[0035] Note that the above-described process that produces a permit for the client code is not strictly necessary, and may be dispensed with in certain situations. If accesses to the service are infrequent, the desired access can simply be performed without giving the client code a permit for successive accesses. Additionally, the permit does not have to be passive. It may include, among other things, a mechanism to inactivate the permit after a certain time period, and a mechanism that maintains a log of uses of the permit. It may also include mechanisms to ensure the permit has not been revoked and to identify users of the permit.

[0036] The foregoing descriptions of embodiments of the invention have been presented for purposes of illustration and description only. They are not intended to be exhaustive or to limit the invention to the forms disclosed. Many modifications and variations will be apparent to practitioners skilled in the art.

Claims

1. A method for providing a first computer program module (122) with an ability to access a service from a second computer program module (112), comprising:

receiving the first computer program module (122);

determining whether the first computer program module (122) has been digitally signed by an authority (204) having power to confer access for the service from the second computer program module (112);

if the first computer program module (122) has been digitally signed by the authority (204) having power to confer access (212, 520) for the service, providing the first computer program module (122) with access to the service; and allowing the first computer program module (122) and the second computer program module (112) to run in the same address space on the same computing node, so that the first computer program module (122) can access the service from the second computer program module (112).

2. The method of claim 1, wherein the act of determining whether the first computer program module (122) has been digitally signed by the authority (204) having power to confer access for the service, includes using a public key associated with the service to verify that the first computer program module (122) has been digitally signed by a corresponding private key (312) for the service.
3. The method of claim 1 or claim 2, wherein the act of determining whether the first computer program module (122) has been digitally signed by the authority having power to confer access for the service, includes verifying that the first computer program module includes a chain of certificates (310) establishing authorisation for the service, a first certificate (312) in the chain (310) being signed by an entity that is originally authorised to confer access for the service, and subsequent certificates (314-316) in the chain (310) being signed by entities (430, 450) that have been delegated authorisation to confer access (516) for the service.
4. The method of any one of claims 1 to 3, wherein the act of providing the first computer program module (122) access to the service, includes providing (212) the first computer program module with a permit that allows the first computer program module (122) to perform a restricted set of services.
5. The method of any one of claims 1 to 4, wherein the service is accessed through an object defined within an object oriented programming system.
6. The method of any one of claims 1 to 5, wherein the first computer program module (122) includes a Java Archive file.

7. The method of any one of claims 1 to 6, wherein the first computer program module (122) includes computer code (320) and at least one digital certificate (312, 314, 316).
8. The method of any one of claims 1 to 7, wherein providing the first computer program module (122) with access to the service allows the first computer program module (122) to interact with the second computer program module (112).
9. The method of claim 8, wherein the first computer program module (122) originates from a first server (120) and is transferred (512) to a computer node (140) for execution, and the second computer program module (112) originates from a second server (110) and is transferred (512) to the computer node (140) for execution.
10. The method of claim 8 or claim 9, wherein the first server (120) and the second server (110) computer are separately located from the computer node (140).
11. The method of any one of claims 1 to 10, wherein the service includes a plurality of services.
12. An apparatus that provides a first computer program module (122) with an ability to access a service from a second computer program module (112), comprising:
 - a computer node (140);
 - a receiving means, within the computer node, that receives the first computer program module (122);
 - a verification means (204), within the computer node, that verifies that the first computer program module (122) has been digitally signed by an authority having power to confer access (212) for the service;
 - an access means (202), within the computer node, that provides the first computer program module (122) with access to the service if the first computer program module has been digitally signed by the authority having power to confer access for the service; and
 - an execution means, within the computer node (140), that allows the first computer program module (122) and the second computer program module (112) to run in the same address space on the same computing node, so that the first computer program module (122) can access the service from the second computer program module (112).
13. The apparatus of claim 12, wherein the verification means (204) is configured to use a public key associated with the service to verify that the first computer program module (122) has been digitally signed by a corresponding private key for the service.
14. The apparatus of claims 12 or 13, wherein the verification means is configured to verify that the first computer program module (122) includes a chain of certificates (310) establishing authorisation for the service, a first certificate (312) in the chain (310) being signed by an entity that is originally authorised to confer access for the service, and subsequent certificates (314-316) in the chain (310) being signed by entities that have been delegated authorisation to confer access for the service.
15. The apparatus of any one of claims 12 to 14, wherein the access means (202) is configured to provide the first computer program module (122) with a permit that allows the first computer program module (122) to perform a restricted set of services.
16. The apparatus of any one of claims 12 to 15, wherein the service is accessed through an object defined within an object oriented programming system.
17. The apparatus of any one of claims 12 to 16, wherein the first computer program module (122) includes a Java Archive file.
18. The apparatus of any one of claims 12 to 17, wherein the first computer program module (122; 432; 434; 436; 452) includes computer code and at least one digital certificate (404; 408; 412; 416; 424).
19. The apparatus of any one of claims 12 to 18, wherein the receiving means is configured to transfer the first computer program module (122) from a first server (120), and to transfer the second computer program module (112) from a second server (110).
20. The apparatus of claim 19, wherein the first server (110) and the second server (120) are separate from the computer node (140).
21. The apparatus of claims 19 or 20, wherein the service includes a plurality of services.
22. A computer readable storage medium storing instructions that when executed by a computer cause the computer to perform a method for providing a first computer program module (122) with an ability to access a service from a second computer program module (112), comprising:
 - receiving the first computer program module

(122);

determining whether the first computer program module (122) has been digitally signed by an authority (400) having power to confer access for the service from the second computer program module (112);

if the first computer program module (122) has been digitally signed by the authority (400) having power to confer access (212, 520) for service, providing the first computer program module with access to the service; and
allowing the first computer program module (122) and the second computer program module (112) to run in the same address space on the same computing node (140), so that the first computer program module (122) can access the service from the second computer program module (112).

23. A computer program module or Java applet (122) which is able to access a service from a second computer program module, comprising;

a computer code section (320), including computer code for execution on a computer node to carry out functions of the first computer program module; and

a digital signature section (310), including a chain of certificates establishing authorization for the service, a first certificate (312) in the chain being signed by an entity that is originally authorized to confer access for the service, and subsequent certificates (314, 316) in the chain being signed by entities that have been delegated authorization to confer access for the service, the digital signature section allowing the computer node to determine whether the computer program module has been granted authority to access the service.

24. A computer program encoding a set of computer instructions for facilitating the provision of a first computer program module (122) with an ability to access a service from a second computer program module (112), which when running on a computer is adapted to perform the method as claimed in any one of the claims 1 to 11.

50

55

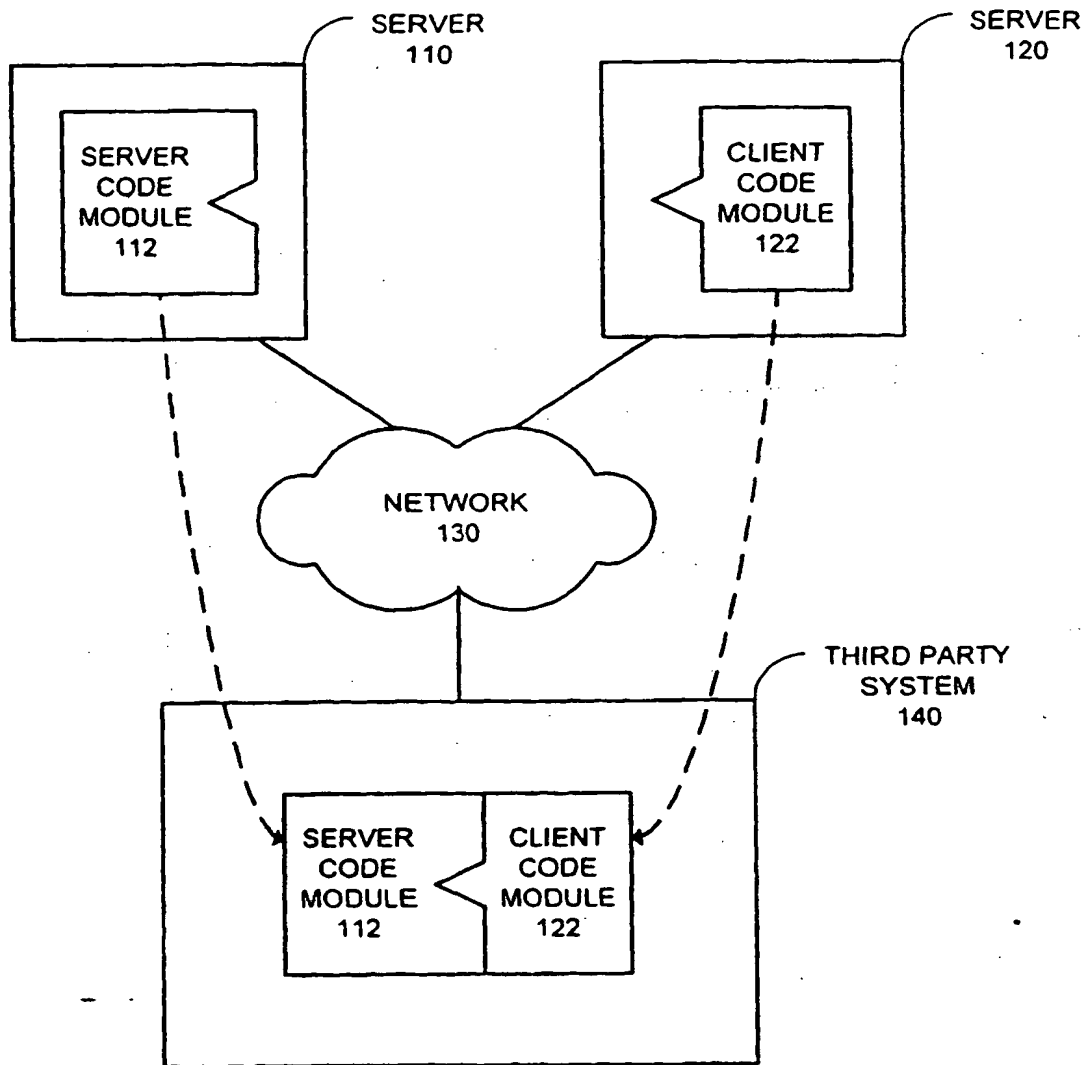


FIG. 1

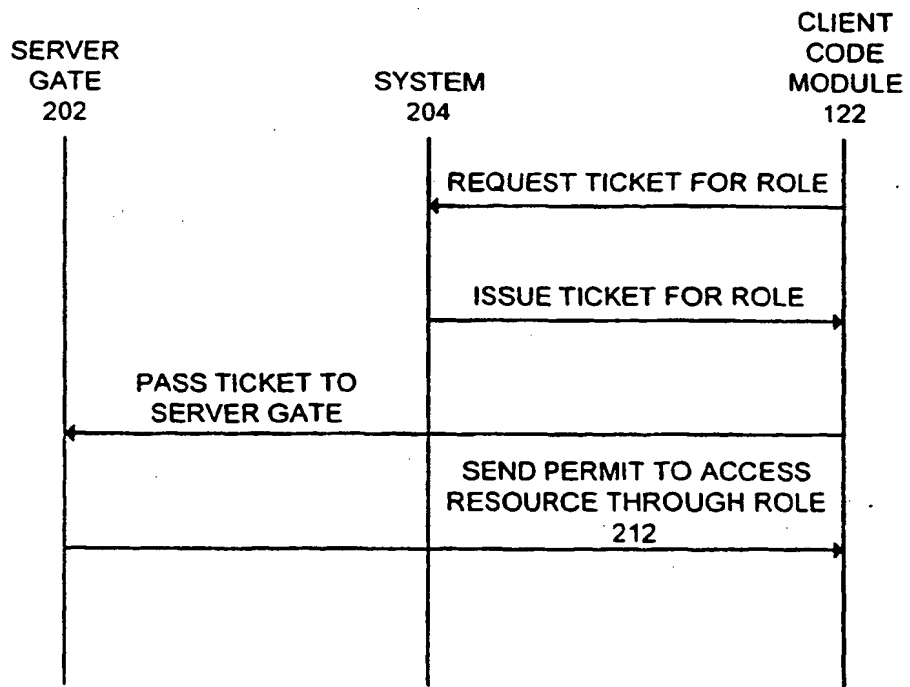


FIG. 2

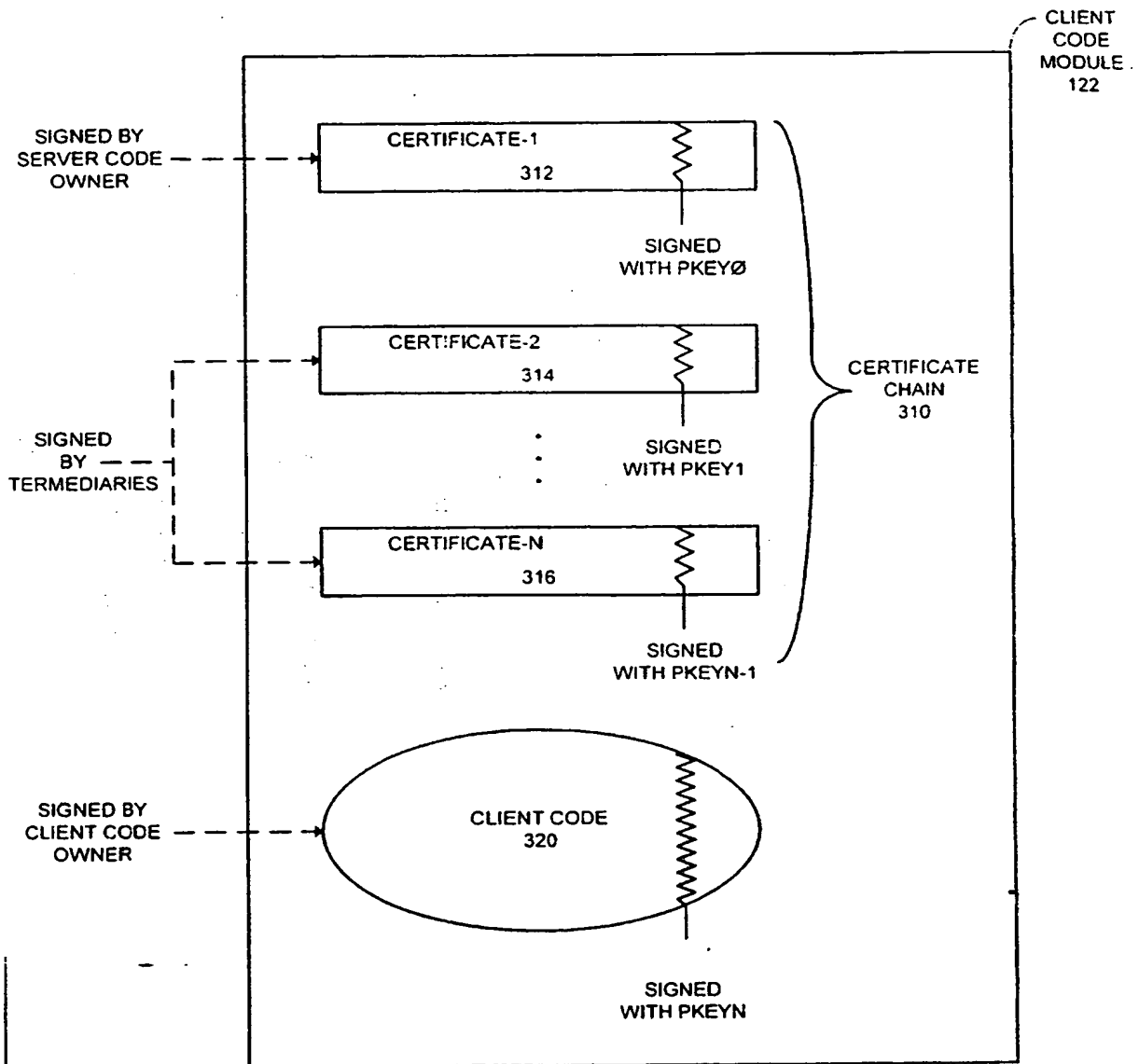
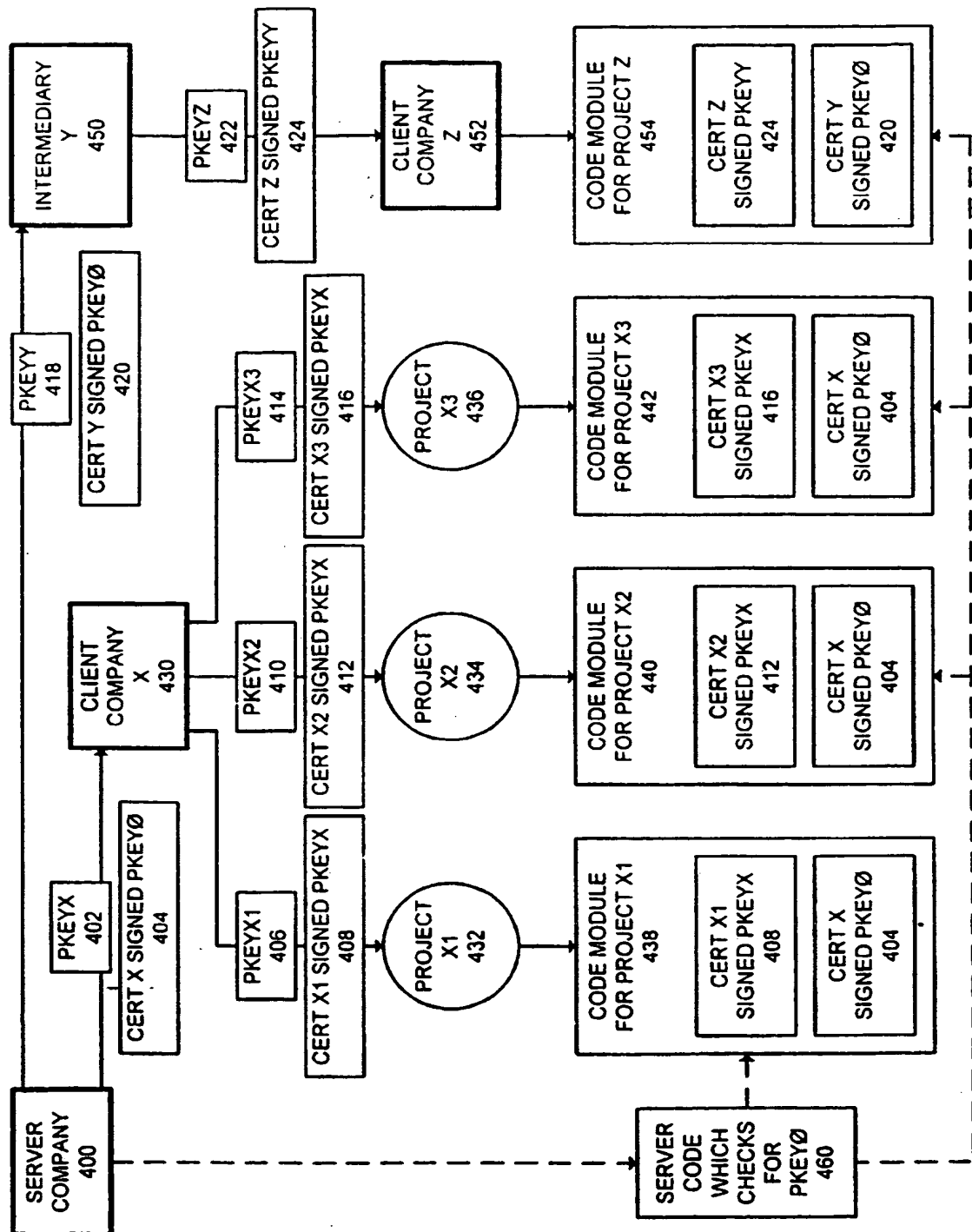


FIG. 3



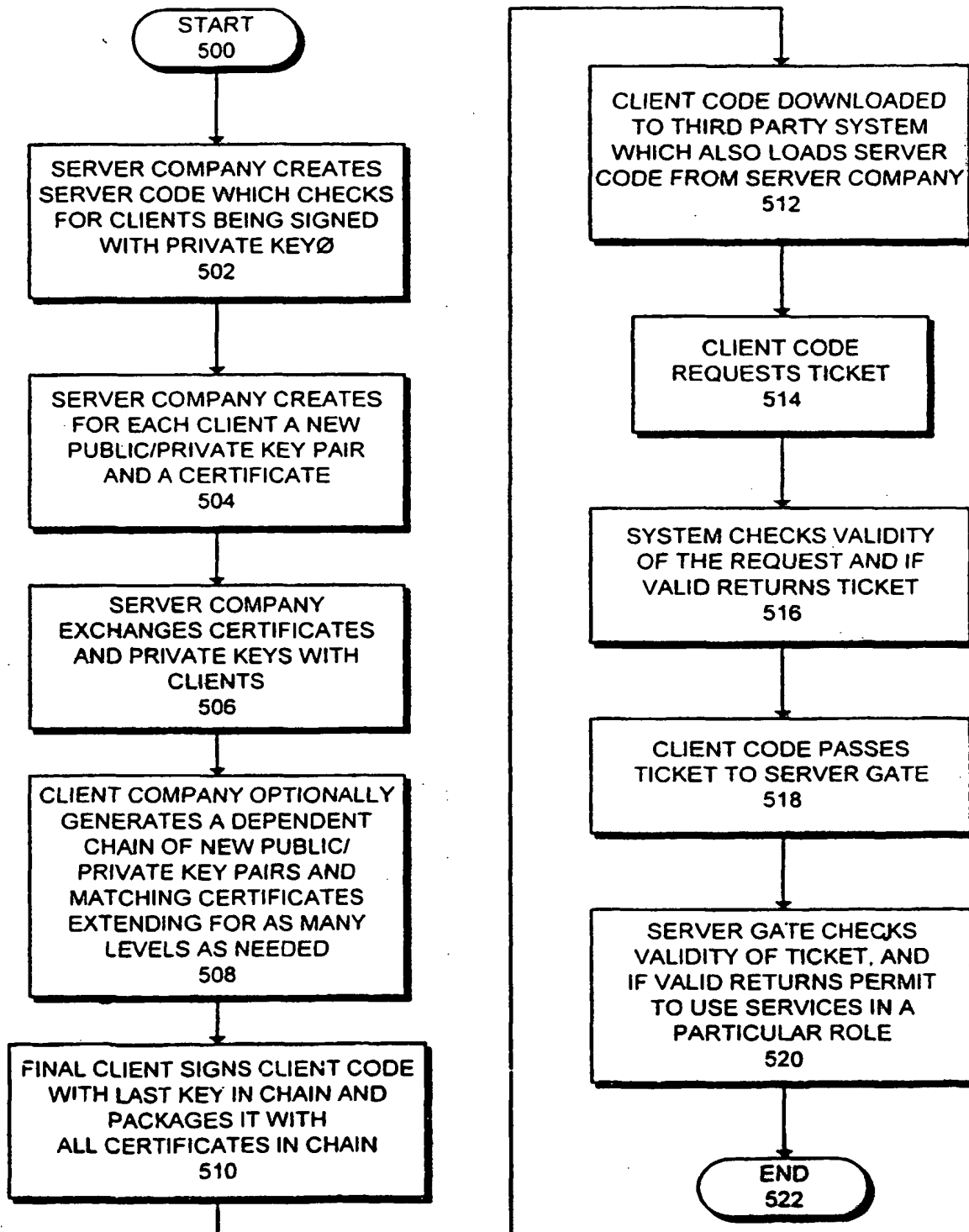


FIG. 5



European Patent
Office

EUROPEAN SEARCH REPORT

Application Number
EP 99 20 2070

DOCUMENTS CONSIDERED TO BE RELEVANT			
Category	Citation of document with indication, where appropriate, of relevant passages	Relevant to claim	CLASSIFICATION OF THE APPLICATION (Int.Cl.7)
Y	WALLACH D S ET AL: "Extensible security architectures for Java" PROCEEDINGS OF THE ACM SYMPOSIUM ON OPERATING SYSTEMS PRINCIPLES, 1997, pages 1-26 14, XP002101681 * page 5, line 7 - page 7, line 11 * * page 8, line 9 - page 10, line 13 * * page 14, line 1 - line 14 * ---	1-24	G06F9/46 G06F1/00
Y	GONG L ET AL: "Going beyond the sandbox: an overview of the new security architecture in the Java/sup TM/ Development Kit 1.2" STAHL UND EISEN, pages 103-112 110, XP002100907 ISSN: 0340-4803 * the whole document * ---	1-24	
A	GRITZALIS S ET AL: "SECURITY ISSUES SURROUNDING PROGRAMMING LANGUAGES FOR MOBILE CODE: JAVA RM VS. SAFE-TCL" OPERATING SYSTEMS REVIEW (SIGOPS), vol. 32, no. 2, 1 April 1998 (1998-04-01), pages 16-32, XP000766954 * page 19, left-hand column, paragraph 1 - page 22, left-hand column, paragraph 4 * * page 26, right-hand column, paragraph 3 - page 27, left-hand column, paragraph 2 * * page 28, left-hand column, paragraph 1 - right-hand column, paragraph 2 * --- -/--	1-24	TECHNICAL FIELDS SEARCHED (Int.Cl.7) G06F
The present search report has been drawn up for all claims			
Place of search THE HAGUE		Date of completion of the search 20 September 1999	Examiner Powell, D
<p>CATEGORY OF CITED DOCUMENTS</p> <p>X : particularly relevant if taken alone Y : particularly relevant if combined with another document of the same category A : technological background O : non-written disclosure P : intermediate document</p> <p>T : theory or principle underlying the invention E : earlier patent document, but published on, or after the filing date D : document cited in the application L : document cited for other reasons & : member of the same patent family, corresponding document</p>			

EPO FORM 1503 03.02 (P4/C01)



European Patent
Office

EUROPEAN SEARCH REPORT

Application Number
EP 99 20 2070

DOCUMENTS CONSIDERED TO BE RELEVANT			
Category	Citation of document with indication, where appropriate, of relevant passages	Relevant to claim	CLASSIFICATION OF THE APPLICATION (Int.Cl.7)
A	N. ISLAM ET AL: "A Flexible Security Model for Using Internet Content" IBM : DEVELOPER : JAVA OVERVIEW : LIBRARY - PAPERS, 'Online! October 1997 (1997-10), XP002115800 IBM Thomas J. Watson Research Center Retrieved from the Internet: <URL:http://www.software.ibm.com/developer/library/flexsecurity/> 'retrieved on 1999-09-17! * page 4 * * introduction; figure 2 * * page 7, paragraph on Java Implementation *	1-24	
A	EP 0 813 133 A (IBM) 17 December 1997 (1997-12-17) * the whole document *	1-24	
A	DIRK BALFANZ ET AL: "Experience with Secure Multi-Processing in Java" TECHNICAL REPORT 560-97, 'Online! May 1998 (1998-05), XP002115807 Department of Computer Science, Princeton University Retrieved from the Internet: <URL:http://www.cs.princeton.edu/sip/pub/1cdcs-final.pdf> 'retrieved on 1999-09-17! * the whole document *		
The present search report has been drawn up for all claims			TECHNICAL FIELDS SEARCHED (Int.Cl.7)
Place of search THE HAGUE		Date of completion of the search 20 September 1999	Examiner Powell, D
CATEGORY OF CITED DOCUMENTS		T : theory or principle underlying the invention E : earlier patent document, but published on, or after the filing date D : document cited in the application L : document cited for other reasons A : technological background O : non-written disclosure P : intermediate document & : member of the same patent family, corresponding document	

EPO FORM 1503 03 02 (Pec01)



European Patent
Office

EUROPEAN SEARCH REPORT

Application Number
EP 99 20 2070

DOCUMENTS CONSIDERED TO BE RELEVANT			
Category	Citation of document with indication, where appropriate, of relevant passages	Relevant to claim	CLASSIFICATION OF THE APPLICATION (Int.Cl.7)
A	BLAZE M ET AL: "DECENTRALIZED TRUST MANAGEMENT" PROCEEDINGS OF THE 1996 IEEE SYMPOSIUM ON SECURITY AND PRIVACY, OAKLAND, CA., MAY 6 - 8, 1996, no. SYMP. 17, 6 May 1996 (1996-05-06), pages 164-173, XP000634842 INSTITUTE OF ELECTRICAL AND ELECTRONICS ENGINEERS ISBN: 0-7803-3527-9 -----		
A	US 5 677 952 A (ROGAWAY PHILLIP W ET AL) 14 October 1997 (1997-10-14) -----		
			TECHNICAL FIELDS SEARCHED (Int.Cl.7)
The present search report has been drawn up for all claims			
Place of search THE HAGUE		Date of completion of the search 20 September 1999	Examiner Powell, D
<p>CATEGORY OF CITED DOCUMENTS</p> <p>X : particularly relevant if taken alone Y : particularly relevant if combined with another document of the same category A : technological background O : non-written disclosure P : intermediate document</p> <p>T : theory or principle underlying the invention E : earlier patent document, but published on, or after the filing date D : document cited in the application L : document cited for other reasons & : member of the same patent family, corresponding document</p>			

EPO FORM 1503 03.82 (P04C01)

**ANNEX TO THE EUROPEAN SEARCH REPORT
ON EUROPEAN PATENT APPLICATION NO.**

EP 99 20 2070

This annex lists the patent family members relating to the patent documents cited in the above-mentioned European search report. The members are as contained in the European Patent Office EDP file on
The European Patent Office is in no way liable for these particulars which are merely given for the purpose of information.

20-09-1999

Patent document cited in search report		Publication date	Patent family member(s)		Publication date
EP 0813133	A	17-12-1997	JP	10091427 A	10-04-1998
US 5677952	A	14-10-1997	US	5454039 A	26-09-1995
			EP	0658022 A	14-06-1995
			JP	7199808 A	04-08-1995
			SG	44363 A	19-12-1997
			US	5675652 A	07-10-1997
			US	5835597 A	10-11-1998

THIS PAGE BLANK (USPTO)